How Does a Bike-Share Navigate Speedy Success?

A case study on R

Prabir Kumar Mitra

In this case study, we will perform a real-world data analysis tasks by incorporating publicly available data.

Scenario

We consider a bike-share company in Chicago: Cyclistic. The director of marketing believes the company's future success depends on maximizing the number of annual memberships. Therefore, our aim is to understand how casual riders and annual members use Cyclistic bikes differently. From these insights, we will design a new marketing strategy to convert casual riders into annual members. But first, Cyclistic executives must approve our recommendations, so they must be backed up with compelling data insights and professional data visualizations.

- Cyclistic's bike-sharing data can be found from the public database. The data has been made available by Motivate International Inc. under this license. We are only including the *Divvy_Trips_2020_Q1.zip* database in our project.

Importing and reading the database

After installing and uploading the file in the current console, we install the following packagaes and call the associated libraries in the working session.

```
library(tidyverse)

## ── Attaching core tidyverse packages ────────────────────────
tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.1     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
## ── Conflicts ──────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
##    Use the conflicted package (<http://conflicted.r-lib.org/>) to
force all conflicts to become errors

library(readr)
library(dplyr)
```

Once these libraries are loaded in the session, we read the databse as follows:

```
raw_tripdata <- read_csv("202104-divvy-tripdata.csv")

## Rows: 337230 Columns: 13
## ── Column specification
────────────────────────────────────────────────────────
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name,
start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## dttm (2): started_at, ended_at
##
##    Use `spec()` to retrieve the full column specification for this
data.
##    Specify the column types or set `show_col_types = FALSE` to
quiet this message.

summary(raw_tripdata)

##    ride_id          rideable_type        started_at

##  Length:337230      Length:337230      Min.   :2021-04-01 00:03:18

##  Class :character   Class :character   1st Qu.:2021-04-07 12:07:56

##  Mode  :character   Mode  :character   Median :2021-04-15 22:37:04

##                                        Mean   :2021-04-15 22:47:10

##                                        3rd Qu.:2021-04-24 08:31:49

##                                        Max.   :2021-04-30 23:59:53

##

##     ended_at                   start_station_name start_station_id

##  Min.   :2021-04-01 00:14:29   Length:337230      Length:337230

##  1st Qu.:2021-04-07 12:31:51   Class :character   Class :character

##  Median :2021-04-15 23:00:10   Mode  :character   Mode  :character
##  Mean   :2021-04-15 23:11:18

##  3rd Qu.:2021-04-24 08:52:47

##  Max.   :2021-05-05 22:14:39

##

##  end_station_name   end_station_id       start_lat       start_lng

##  Length:337230      Length:337230      Min.   :41.64   Min.   :-
87.78
##  Class :character   Class :character   1st Qu.:41.88   1st Qu.:-
87.66
##  Mode  :character   Mode  :character   Median :41.90   Median :-
87.64
##                                        Mean   :41.90   Mean   :-
87.64
##                                        3rd Qu.:41.93   3rd Qu.:-
87.63
##                                        Max.   :42.07   Max.   :-
87.52
##

##     end_lat         end_lng       member_casual
##  Min.   :41.59   Min.   :-87.85   Length:337230
##  1st Qu.:41.88   1st Qu.:-87.66   Class :character
##  Median :41.90   Median :-87.64   Mode  :character
##  Mean   :41.90   Mean   :-87.65
##  3rd Qu.:41.93   3rd Qu.:-87.63
##  Max.   :42.15   Max.   :-87.52
##  NA's   :267     NA's   :267
```

In our session, we assign the name ‘tripdata’ to the database. We readily notice that the database has 13 columns and 337230 rows. Our next strategy is to clean the database that includes removing possible duplication, entries with missing values and impractical (mistakenly) entries. ## Tidying and orgnising the database

```
tripdata <- na.omit(raw_tripdata)
tripdata <- distinct(tripdata)
summary(tripdata)
```

```
##    ride_id           rideable_type        started_at

##  Length:298207      Length:298207      Min.   :2021-04-01 00:03:18

##  Class :character   Class :character   1st Qu.:2021-04-07 09:09:11
##  Mode  :character   Mode  :character   Median :2021-04-15 18:37:56
##                                          Mean   :2021-04-15 20:09:24
##                                          3rd Qu.:2021-04-24 00:46:14
##                                          Max.   :2021-04-30 23:59:53
##     ended_at                   start_station_name start_station_id
##  Min.   :2021-04-01 00:14:29   Length:298207      Length:298207
##  1st Qu.:2021-04-07 09:30:50   Class :character   Class :character
##  Median :2021-04-15 18:54:15   Mode  :character   Mode  :character
##  Mean   :2021-04-15 20:33:25
##  3rd Qu.:2021-04-24 01:05:31
##  Max.   :2021-05-05 22:14:39
##  end_station_name   end_station_id       start_lat       start_lng
##  Length:298207      Length:298207      Min.   :41.65   Min.   :-
87.77
##  Class :character   Class :character   1st Qu.:41.88   1st Qu.:-
87.66
##  Mode  :character   Mode  :character   Median :41.90   Median :-
87.64
##                                          Mean   :41.90   Mean   :-
87.64
##                                          3rd Qu.:41.93   3rd Qu.:-
87.63
##                                          Max.   :42.06   Max.   :-
87.53
##     end_lat         end_lng       member_casual
##  Min.   :41.65   Min.   :-87.77   Length:298207
##  1st Qu.:41.88   1st Qu.:-87.66   Class :character
##  Median :41.90   Median :-87.64   Mode  :character
##  Mean   :41.90   Mean   :-87.64
##  3rd Qu.:41.93   3rd Qu.:-87.63
##  Max.   :42.06   Max.   :-87.53
```

We readily notice that the number of rows has reduced to 298207 from 337230.

We want to add three columns to the databse now:

1. duration of the trips in Hours
2. days of the week when those trips were made. For easy and better calculation, we will assign numerical values 1, 2, 3 … to the weekdays as: Sunday -> 1, Monday -> 2, Tuesday -> 3, etc.
3. length of the trips. A zero length implies that the trips started and ended at the same station.

```
tripdata$duration <- as.numeric(difftime(tripdata$ended_at,
tripdata$started_at,units="hours"))
tripday=weekdays(tripdata$started_at)
dates=c("Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday")
tripday <- as.integer(factor(tripday, levels = dates,ordered = TRUE))
tripdata$tripday <- tripday
tripdata$length <- sqrt((tripdata$start_lat-tripdata$end_lat)^2+
(tripdata$start_lng-tripdata$end_lng)^2)
summary(tripdata)
```

```
##    ride_id          rideable_type        started_at                 
##  Length:298207      Length:298207      Min.   :2021-04-01 00:03:18
##  Class :character   Class :character   1st Qu.:2021-04-07 09:09:11
##  Mode  :character   Mode  :character   Median :2021-04-15 18:37:56
##                                        Mean   :2021-04-15 20:09:24
##                                        3rd Qu.:2021-04-24 00:46:14
##                                        Max.   :2021-04-30 23:59:53
##                                                                  
##     ended_at                   start_station_name start_station_id
##  Min.   :2021-04-01 00:14:29   Length:298207      Length:298207
##  1st Qu.:2021-04-07 09:30:50   Class :character   Class :character
##  Median :2021-04-15 18:54:15   Mode  :character   Mode  :character
##  Mean   :2021-04-15 20:33:25
##  3rd Qu.:2021-04-24 01:05:31
##  Max.   :2021-05-05 22:14:39
##
```

```
##  end_station_name   end_station_id       start_lat       start_lng

##  Length:298207      Length:298207      Min.   :41.65   Min.   :-
87.77
##  Class :character   Class :character   1st Qu.:41.88   1st Qu.:-
87.66
##  Mode  :character   Mode  :character   Median :41.90   Median :-
87.64
##                                        Mean   :41.90   Mean   :-
87.64
##                                        3rd Qu.:41.93   3rd Qu.:-
87.63
##                                        Max.   :42.06   Max.   :-
87.53
##

##     end_lat         end_lng       member_casual         duration

##  Min.   :41.65   Min.   :-87.77   Length:298207      Min.   : -
0.0028
##  1st Qu.:41.88   1st Qu.:-87.66   Class :character   1st Qu.:
0.1206
##  Median :41.90   Median :-87.64   Mode  :character   Median :
0.2150
##  Mean   :41.90   Mean   :-87.64                      Mean   :
0.4004
##  3rd Qu.:41.93   3rd Qu.:-87.63                      3rd Qu.:
0.4003
##  Max.   :42.06   Max.   :-87.53
Max.   :796.2783
##

##     tripday           length
##  Min.   : NA      Min.   :0.000000
##  1st Qu.: NA      1st Qu.:0.008882
##  Median : NA      Median :0.016520
##  Mean   :NaN      Mean   :0.021216
##  3rd Qu.: NA      3rd Qu.:0.028689
##  Max.   : NA      Max.   :0.276268
##  NA's   :298207
```

Here, we notice that the minimum duration is negative which can not be accepted. Therefore, we will remove all the entries from the database where end time (*ended_at*) is noted to be before the start time (*started_at*).

```
tripdata %>%  filter(duration < 0)

## # A tibble: 4 × 16
##   ride_id          rideable_type started_at          ended_at
##   <chr>            <chr>         <dttm>              <dttm>

## 1 BC53ECCBC76278FD classic_bike  2021-04-07 16:11:33 2021-04-07
16:11:26
## 2 6E81034B446FC2FD electric_bike 2021-04-23 09:43:39 2021-04-23
09:43:29
## 3 318DD838369AEA61 classic_bike  2021-04-30 10:56:32 2021-04-30
10:56:30
## 4 8ADD13BD8F6A7567 classic_bike  2021-04-17 12:43:36 2021-04-17
12:43:27
## #    12 more variables: start_station_name <chr>, start_station_id
<chr>,
## #   end_station_name <chr>, end_station_id <chr>, start_lat <dbl>,
## #   start_lng <dbl>, end_lat <dbl>, end_lng <dbl>, member_casual
<chr>,
## #   duration <dbl>, tripday <int>, length <dbl>
```

```
tripdata <- tripdata %>% filter(duration > 0)
summary(tripdata)
```

```
##    ride_id          rideable_type        started_at

##  Length:298199      Length:298199      Min.   :2021-04-01 00:03:18

##  Class :character   Class :character   1st Qu.:2021-04-07 09:09:05

##  Mode  :character   Mode  :character   Median :2021-04-15 18:37:29

##                                        Mean   :2021-04-15 20:09:07

##                                        3rd Qu.:2021-04-24 00:45:53

##                                        Max.   :2021-04-30 23:59:53

##

##     ended_at                   start_station_name start_station_id

##  Min.   :2021-04-01 00:14:29   Length:298199      Length:298199

##  1st Qu.:2021-04-07 09:30:29   Class :character   Class :character

##  Median :2021-04-15 18:54:09   Mode  :character   Mode  :character

##  Mean   :2021-04-15 20:33:09

##  3rd Qu.:2021-04-24 01:05:23
##  Max.   :2021-05-05 22:14:39

##

##  end_station_name   end_station_id       start_lat       start_lng

##  Length:298199      Length:298199      Min.   :41.65   Min.   :-
87.77
##  Class :character   Class :character   1st Qu.:41.88   1st Qu.:-
87.66
##  Mode  :character   Mode  :character   Median :41.90   Median :-
87.64
##                                        Mean   :41.90   Mean   :-
87.64
##                                        3rd Qu.:41.93   3rd Qu.:-
87.63
##                                        Max.   :42.06   Max.   :-
87.53
##

##     end_lat         end_lng       member_casual         duration

##  Min.   :41.65   Min.   :-87.77   Length:298199      Min.   :
0.0003
##  1st Qu.:41.88   1st Qu.:-87.66   Class :character   1st Qu.:
0.1206
##  Median :41.90   Median :-87.64   Mode  :character   Median :
0.2150
##  Mean   :41.90   Mean   :-87.64                      Mean   :
0.4004
##  3rd Qu.:41.93   3rd Qu.:-87.63                      3rd Qu.:
0.4003
##  Max.   :42.06   Max.   :-87.53
Max.   :796.2783
##

##     tripday           length
##  Min.   : NA      Min.   :0.000000
##  1st Qu.: NA      1st Qu.:0.008884
##  Median : NA      Median :0.016520
##  Mean   :NaN      Mean   :0.021216
##  3rd Qu.: NA      3rd Qu.:0.028691
##  Max.   : NA      Max.   :0.276268
##  NA's   :298199
```

Following this, the number of rows in the database has further reduced to 298199. Let us now look at trip durations in detail, in relation with member status and bike type.

```
bike_type <- table(tripdata$member_casual,tripdata$rideable_type)
bike_type_data <- data.frame(member_type=c("casual","member"),
classic_bike=bike_type[,1],docked_bike=bike_type[,2],electric_bike=bik
e_type[,3])
bike_type_data <- rbind(bike_type_data,
list("Total",sum(bike_type_data$classic_bike),
sum(bike_type_data$docked_bike), sum(bike_type_data$electric_bike)))
btt <- data.frame("member_type"=bike_type_data$member_type,
"classic_bike"=bike_type_data$classic_bike,
"docked_bike"=bike_type_data$docked_bike,
"electric_bike"=bike_type_data$electric_bike)
btt$total <- rowSums(btt[,2:4])
bike_type <- btt
bike_type

##   member_type classic_bike docked_bike electric_bike  total
## 1      casual        70502       24713         25203 120418
## 2      member       143621           0         34160 177781
## 3       Total       214123       24713         59363 298199
```

It is clear from the above table that the number of annual members is ~1.5 times the number of casual members. Among the different types of bikes, classic bikes are much more in use than other types. However, very interestingly, the docked type bikes are only used by casual members. We look at the average duration of ridership in the next table.

```
tripdata %>%
  group_by(member_casual) %>%
  summarise(avg_duration=mean(duration))

## # A tibble: 2 × 2
##   member_casual avg_duration
##   <chr>                <dbl>
## 1 casual               0.641
## 2 member               0.238

tripdata %>%
  group_by(member_casual,rideable_type) %>%
  summarise(avg_duration=mean(duration))

## `summarise()` has grouped output by 'member_casual'. You can
override using the
## `.groups` argument.

## # A tibble: 5 × 3
## # Groups:   member_casual [2]
##   member_casual rideable_type avg_duration
##   <chr>         <chr>                <dbl>
## 1 casual        classic_bike         0.479
## 2 casual        docked_bike          1.39
## 3 casual        electric_bike        0.357
## 4 member        classic_bike         0.240
## 5 member        electric_bike        0.226
```

As we find, despite the members being much more in number than the casual riders, the average duration of rides is almost 3 times than that of the members. Further, the ridership with docked bike is extremely high compared to other types of bikes. Here we remeber that docked type bikes are used by casual members only.

```
d_bike <- tripdata %>%
  filter(rideable_type == "docked_bike") %>%
  group_by(tripday) %>%
  summarise(avg_duration=mean(duration))
d_bike
```

```
## # A tibble: 1 × 2
##   tripday avg_duration
##     <int>        <dbl>
## 1      NA         1.39
```

Here we immediately notice that, while average ridership in docked bikes are similar in most of the weekdays, except for Thursday and Friday. On Thursday, the ridership seems to be lowest and on Friday the ridership seems to be the most. On both of these two days, the ridership average is extremely low or high compared to other days. Therefore, we look into details of these two days ridership on docked bikes.

```
tripdata %>%
  filter(rideable_type == "docked_bike" & tripday == 5) %>%
  group_by(start_station_name,end_station_name) %>%
  summarise(avg_duration=mean(duration)) %>%
  arrange(desc(avg_duration))
```

```
## `summarise()` has grouped output by 'start_station_name'. You can
override
## using the `.groups` argument.
```

```
## # A tibble: 0 × 3
## # Groups:   start_station_name [0]
## #    3 variables: start_station_name <chr>, end_station_name <chr>,
## #   avg_duration <dbl>
```

Clearly, there is one exceptional entry of ridership of 34.5 hours on Thursday, whereas all the other entries are of less than 25 hours. This particular ride initiated and terminated both at the “Chicago State University (CSU)”. In the following, we look at the specific entries from trips: CSU -> CSU on Thursday as well as on each weekdays.

```
csu <- "Chicago State University"
tripdata %>%
  filter(rideable_type == "docked_bike" & start_station_name == csu &
end_station_name == csu & tripday == 5)
```

```
## # A tibble: 0 × 16
## #    16 variables: ride_id <chr>, rideable_type <chr>, started_at
<dttm>,
## #   ended_at <dttm>, start_station_name <chr>, start_station_id
<chr>,
## #   end_station_name <chr>, end_station_id <chr>, start_lat <dbl>,
## #   start_lng <dbl>, end_lat <dbl>, end_lng <dbl>, member_casual
<chr>,
## #   duration <dbl>, tripday <int>, length <dbl>
```

```
tripdata %>%
  filter(rideable_type == "docked_bike" & start_station_name == csu &
end_station_name == csu) %>%
  group_by(tripday) %>%
  summarise(duration) %>%
  arrange(desc(duration))
```

```
## Warning: Returning more (or less) than 1 row per `summarise()`
group was deprecated in
## dplyr 1.1.0.
##    Please use `reframe()` instead.
##    When switching from `summarise()` to `reframe()`, remember that
`reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this
warning was
## generated.
```

```
## `summarise()` has grouped output by 'tripday'. You can override
using the
## `.groups` argument.
```

```
## # A tibble: 9 × 2
## # Groups:   tripday [1]
##   tripday duration
##     <int>    <dbl>
## 1      NA   68.8
## 2      NA    1.28
## 3      NA    1.26
## 4      NA    1.17
## 5      NA    0.944
## 6      NA    0.658
## 7      NA    0.246
## 8      NA    0.218
## 9      NA    0.210
```

Strangely, while the total ridership from CSU to CSU on other weekdays are only a few minutes to less than two hours, one particular entry with the *ride_id*: 5D0B0CCDB4238065 stands out as it had a duration of more than 60 hours.

Next, we look into details of the docked bike rides on Fridays.

```
tripdata %>%
  filter(rideable_type == "docked_bike" & tripday == 6) %>%
  group_by(ride_id,started_at,ended_at) %>%
  summarise(duration) %>%
  arrange(desc(duration))

## Warning: Returning more (or less) than 1 row per `summarise()`
group was deprecated in
## dplyr 1.1.0.
##    Please use `reframe()` instead.
##    When switching from `summarise()` to `reframe()`, remember that
`reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this
warning was
## generated.

## `summarise()` has grouped output by 'ride_id', 'started_at',
'ended_at'. You
## can override using the `.groups` argument.

## # A tibble: 0 × 4
## # Groups:   ride_id, started_at, ended_at [0]
## #    4 variables: ride_id <chr>, started_at <dttm>, ended_at
<dttm>,
## #   duration <dbl>
```

Here, one particular entry with the *ride_id*: E84DF812305C9C9F stands out in view of its unusually long duration. After excluding this entry from the calculation, the average ridership on Friday reduces to 1.83 hours which is comparable to the other weekdays except Thursday.

```
f_tripdata <- tripdata %>%
  filter(rideable_type == "docked_bike" & tripday == 6)

f_tripdata %>%
  filter(duration < 796.) %>%
  summarise(mean(duration))

## # A tibble: 1 × 1
##   `mean(duration)`
##              <dbl>
## 1              NaN
```

Observations

In this section, we will summarise our key observations from the analysis. They are listed below:

1. Number of annual members are ~1.5 times than that of casual riders. However, casual riders use Cyclistic's bikes almost 3 times than that of annual riders.

2. Cyclistic offers three type of bikes: classical, docked and electric. While classical and electric bikes are used by both member and casual type riders, the docked bike seems to be used by only the casual riders.

3. Docked bikes, although least in number compared to other two types of bikes, are used for the longest rides on average. Remebering that this type of bikes are only used by casual riders, an assessment can be made that casual riders hire docked bikes for spending leisure time i.e., for short trips over the weekends etc. This argument is further supported by the fact that the longest ridership on average initiated on Fridays.

4. A couple of entries are worth re-checking, which could not be possible in this project. These are:

- *ride_id*: 5D0B0CCDB4238065 (CSU -> CSU, duration: ~69 Hours)
- *ride_id*: E84DF812305C9C9F (Duration: 796 Hours)

Conclusions and recommendations

It should be remembered here that the dataset does not provide the membership amount as well as casual ride prices. Without these values, it becomes difficult to explain the disparity between the number of membership and ridership. However, from the analysis, we can make one recommendation with certainty.

Considering, docked bikes are mostly used for leisure activities, Cyclistic can provide lucrative offers to sell memberships for the leisure riders. A few recommendations include:

- A limited number of free rides can be provided upon completing a certian number of leisure rides.
- Cyclistic can add live tracking, intimation of fun activities, locations of convenient stores, restrooms etc. on their app which will make leisure rides more convenient. If these features are only made available with a membership then the number of membership can be significantly increased.

- Some discount on the membership can be provided for rides who use the bikes to commute to universities or other public services.